

# LAPORAN TUGAS PEMROGRAMAN BERBASIS FRAMEWORK

Modul 4  
Interaksi dengan API

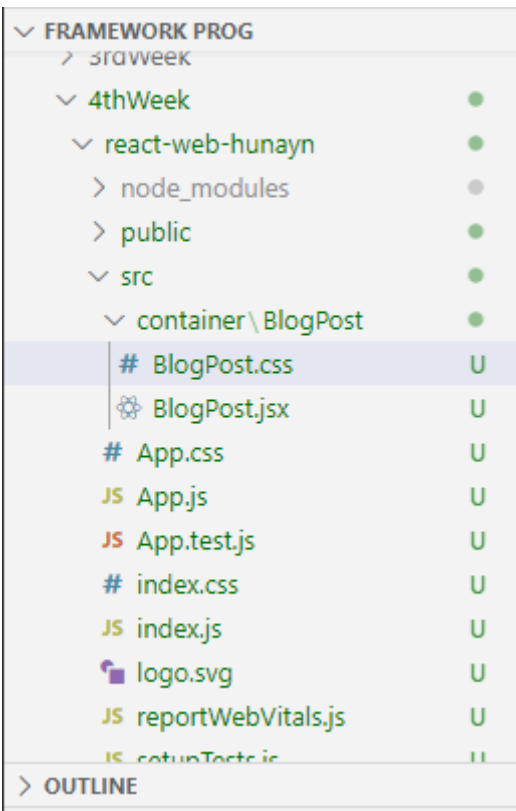


Disusun Oleh :

Nama : Hunayn Risatayn  
NIM : 1841720148  
Kelas : TI-3E

Program Studi D-IV Teknik Informatika  
Jurusan Teknologi Informasi  
Politeknik Negeri Malang  
Maret 2021

## Praktikum – Bagian 1: Interaksi dengan API menggunakan method GET

Langkah	Keterangan
1	<p>Buka Project React pada pertemuan sebelumnya dan jalankan “npm start” menggunakan cmd dalam direktori tersebut.</p> <pre> PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek&gt; npx create-react-app react-web-hunayn npx: installed 67 in 7.979s  Creating a new React app in D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn.  Installing packages. This might take a couple of minutes. Installing react, react-dom, and react-scripts with cra-template...  &gt; core-js@2.6.12 postinstall D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn\node_modules\babel-runtim e\node_modules\core-js &gt; node -e "try{require('./postinstall')}catch(e){}"  npm run eject   Removes this tool and copies build dependencies, configuration files   and scripts into the app directory. If you do this, you can't go back!  We suggest that you begin by typing:    cd react-web-hunayn   npm start  Happy hacking! PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek&gt; </pre> <p>Buat folder baru bernama “BlogPost” pada folder container (statefull component).</p> <pre> PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn\src\container&gt; mkdir BlogPost  Directory: D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn\src\container  Mode                LastWriteTime         Length Name ----                - d-----          3/1/2021   7:40 AM             BlogPost  PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn\src\container&gt; </pre> <p>Buat file BlogPost.jsx dan BlogPost.css di dalam folder “BlogPost”, seperti pada Gambar 1.2.</p> 

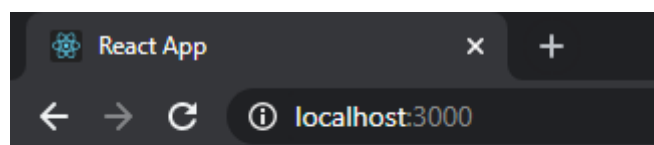
Buka file BlogPost.jsx dan ketikkan kode seperti Gambar 1.3.

```
framework_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > default
1  import React, {Component} from "react";
2  import './BlogPost.css';
3
4  class BlogPost extends Component {
5    render() {
6      return (
7        <p>Blog Artikel</p>
8      )
9    }
10 } export default BlogPost;
```

Pada file index.js, lakukan import component BlogPost seperti Gambar 1.4

```
framework_prog > 4thWeek > react-web-hunayn > src > JS index.js
1  import React from "react";
2  import ReactDOM from "react-dom";
3  import './index.css';
4  // import App from './App';
5  // import reportWebVitals from './reportWebVitals';
6  import * as serviceWorker from './serviceWorker';
7
8  import BlogPost from './container/BlogPost/BlogPost';
9
10 ReactDOM.render(
11   // <React.StrictMode>
12   //   <App />
13   // </React.StrictMode>
14   <BlogPost />,
15   document.getElementById("root")
16 );
17
18 // If you want to start measuring performance in your app, pass a function
19 // to log results (for example: reportWebVitals(console.log))
20 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
21
22 // reportWebVitals();
23 serviceWorker.unregister();
24
```

Pada web browser akan tampil seperti pada Gambar 1.5.



Blog Artikel

Tahapan selanjutnya adalah perbaikan tampilan sebuah website untuk mempercantik halaman website tersebut dengan menggunakan Bootstrap yang umum digunakan.

Import css bootstrap.min.css (css bootstrap yang sudah dikompresi) ke dalam index.js (seperti Gambar 1.6). Jika css tidak ditemukan, install lewat cmd dengan perintah “npm install bootstrap”

```
PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn> npm install bootstrap
npm WARN tsutils@3.20.0 requires a peer of typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta but none is installed. You must install peer dependencies yourself.
npm WARN bootstrap@4.6.0 requires a peer of jquery@1.9.1 - 3 but none is installed. You must install peer dependencies yourself.
npm WARN bootstrap@4.6.0 requires a peer of popper.js@^1.16.1 but none is installed. You must install peer dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.3.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\watchpack-chokidar2\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\webpack-dev-server\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
```

```
framework_prog > 4thWeek > react-web-hunayn > src > JS index.js
1 import React from "react";
2 import ReactDOM from "react-dom";
3 import "bootstrap/dist/css/bootstrap.min.css";
4 import "./index.css";
5 // import App from "./App";
6 // import reportWebVitals from './reportWebVitals';
7 import * as serviceWorker from "./serviceWorker";
8
9 import BlogPost from "../container/BlogPost/BlogPost";
10
11 ReactDOM.render(
12   // <React.StrictMode>
13   // <App />
14   // </React.StrictMode>
15   <BlogPost />,
16   document.getElementById("root")
17 );
18
19 // If you want to start measuring performance in your app, pass a function
20 // to log results (for example: reportWebVitals(console.log))
21 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
22
23 // reportWebVitals();
24 serviceWorker.unregister();
25
```

Modifikasi file index.html pada folder "public" seperti Gambar 1.7. Cermati code program yang ada dalam gambar!.

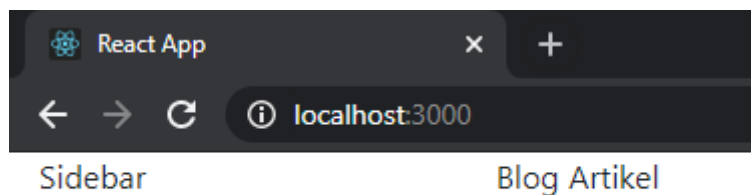
```
framework_prog > 4thWeek > react-web-hunayn > public > <> index.html > html > body
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6     <meta name="viewport" content="width=device-width, initial-scale=1" />
7     <meta name="theme-color" content="#000000" />
8     <meta
9       name="description"
10      content="Web site created using create-react-app"
11    />
12    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13    <!--
14      manifest.json provides metadata used when your web app is installed on a
15      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
16    -->
17    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
18    <!--
19      Notice the use of %PUBLIC_URL% in the tags above.
20      It will be replaced with the URL of the `public` folder during the build.
21      Only files inside the `public` folder can be referenced from the HTML.
22
23      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
24      work correctly both with client-side routing and a non-root public URL.
25      Learn how to configure a non-root public URL by running `npm run build`.
26    -->
27    <title>React App</title>
28  </head>
```

```

28 </head>
29 <body>
30 <noscript>You need to enable JavaScript to run this app.</noscript>
31 <!-- <div id="root"></div> -->
32 <!--
33   This HTML file is a template.
34   If you open it directly in the browser, you will see an empty page.
35
36   You can add webfonts, meta tags, or analytics to this file.
37   The build step will place the bundled scripts into the <body> tag.
38
39   To begin the development, run `npm start` or `yarn start`.
40   To create a production bundle, use `npm run build` or `yarn build`.
41 -->
42 <div class="container-fluid">
43   <div class="row">
44     <div class="col-2" id="sidebar">Sidebar</div>
45     <div class="col-10" id="content">Blog Artikel</div>
46   </div>
47 </div>
48 </body>
49 </html>
50

```

Amati tampilan yang ada pada browser (seperti Gambar 1.8)



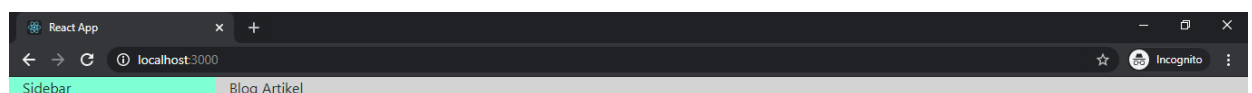
Buka file index.css dan tambahkan code css seperti Gambar 1.9, untuk menambah sedikit style pada halaman web

```

framework_prog > 4thWeek > react-web-hunayn > src > # index.css > #content
1  body {
2    margin: 0;
3    font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", "Roboto", "Oxygen",
4      "Ubuntu", "Cantarell", "Fira Sans", "Droid Sans", "Helvetica Neue",
5      sans-serif;
6    -webkit-font-smoothing: antialiased;
7    -moz-osx-font-smoothing: grayscale;
8  }
9
10 code {
11   font-family: source-code-pro, Menlo, Monaco, Consolas, "Courier New",
12     monospace;
13 }
14
15 #sidebar {
16   background-color: aquamarine;
17 }
18
19 #content {
20   background-color: lightgray;
21 }
22

```

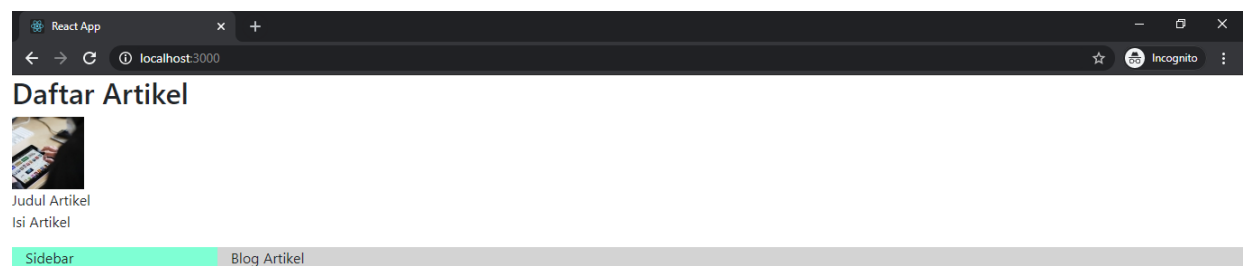
Perhatikan kembali browser, dan lihat hasil tampilan seperti Gambar 1.10.



Kita ingin sebuah website memiliki tampilan seperti pada Gambar 1.1. Dengan minimal ada gambar artikel, judul, dan deskripsi artikel. Maka contoh data dummy yang akan kita pakai bisa menggunakan data dari <http://placeimg.com> contoh <http://placeimg.com/120/120/any>. Tahapan edit tampilan post artikel:

## Ubah kode program untuk statefull component BlogPost.jsx menjadi seperti Gambar 1.11

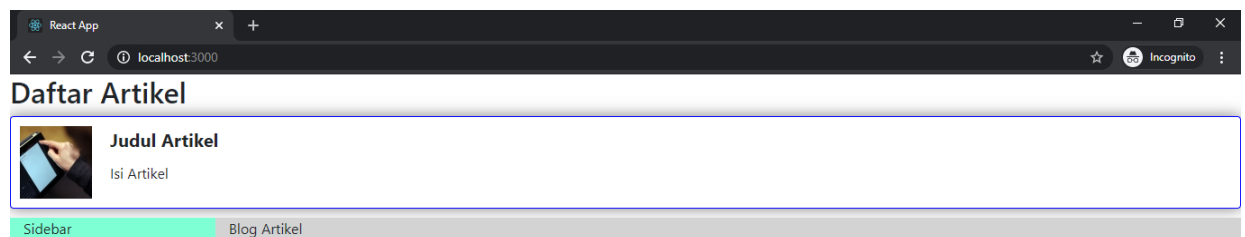
```
framework_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > BlogPost > render
1  import React, {Component} from "react";
2  import './BlogPost.css';
3
4  class BlogPost extends Component {
5    render() {
6      return (
7        <div class="post-artikel">
8          <h2>Daftar Artikel</h2>
9          <div class="artikel">
10             <div class="gambar-artikel">
11               
12             </div>
13             <div class="konten-artikel">
14               <div class="judul-artikel">Judul Artikel</div>
15               <p class="isi-artikel">Isi Artikel</p>
16             </div>
17           </div>
18         </div>
19       );
20     }
21   } export default BlogPost;
```



## Tambahkan custom css ke BlogPost.css seperti Gambar 1.12

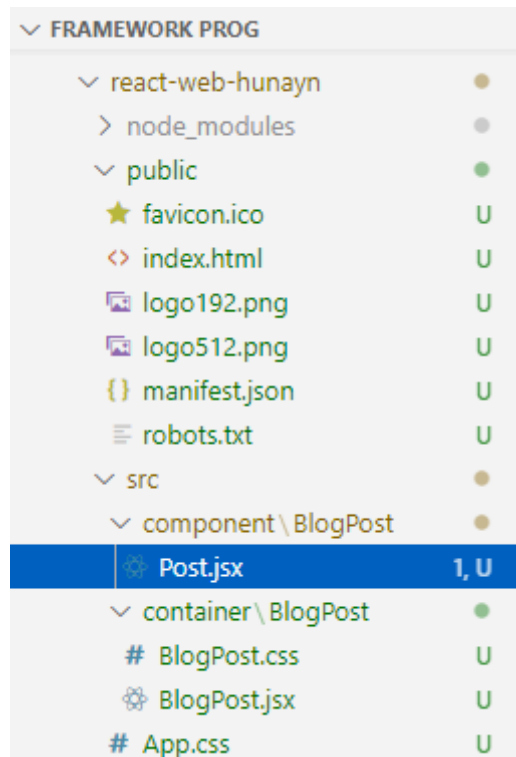
```
framework_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > # BlogPost.css > .konten-artikel p.isi-artikel
1  .artikel {
2    width: 100%;
3    padding: 10px;
4    border: 1px solid blue;
5    border-radius: 4px;
6    margin-bottom: 10px;
7    box-shadow: 0 0 16px rgba(0, 0, 0, 0.5);
8    display: flex;
9  }
10
11  .gambar-artikel {
12    height: 80px;
13    width: 80px;
14    margin-right: 20px;
15    vertical-align: top;
16  }
17
18  .gambar-artikel img {
19    width: 100%;
20    height: 100%;
21    object-fit: cover;
22  }
23
24  .konten-artikel {
25    flex: 1;
26  }
27
28  .konten-artikel {
29    flex: 1;
30  }
31
32  .konten-artikel div.judul-artikel {
33    font-size: 20px;
34    font-weight: bold;
35    margin-bottom: 10px;
36  }
37
38  .konten-artikel p.isi-artikel {
39    font-size: 16px;
40    margin-bottom: 10px;
41  }
```

Perhatikan tampilan browser.



### Pemindahan dari statefull component ke stateless component

Buat folder BlogPost pada folder component (stateless component), lalu buat file Post.jsx



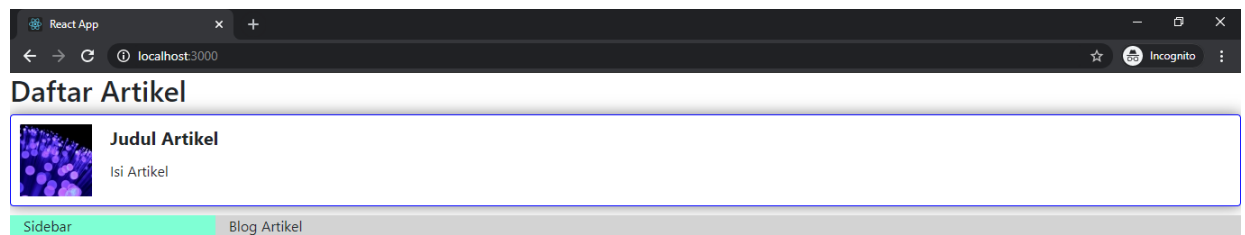
Potong (cut) baris 9-17 pada statefull component BlogPost.jsx ke stateless component Post.jsx, dan modifikasi Post.jsx seperti Gambar 1.13.



Untuk statefull component BlogPost.jsx pada baris 10, panggil stateless component Post.jsx seperti Gambar 1.14.

```
framework_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > default
1  import React, {Component} from "react";
2  import './BlogPost.css';
3  import Post from "../../component/BlogPost/Post";
4
5  class BlogPost extends Component {
6    render() {
7      return (
8        <div class="post-artikel">
9          <h2>Daftar Artikel</h2>
10         <Post />
11       </div>
12     )
13   }
14 } export default BlogPost;
```

Perhatikan hasil tampilan browser, apa yang terjadi?



### Muat Data Dinamis.

Pada statefull component BlogPost.jsx, tambahkan parameter yang ingin dilempar ke stateless component untuk ditampilkan. Kode program bisa dilihat pada Gambar 1.15.

```
framework_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > BlogPost > render
1  import React, {Component} from "react";
2  import './BlogPost.css';
3  import Post from "../../component/BlogPost/Post";
4
5  class BlogPost extends Component {
6    render() {
7      return (
8        <div class="post-artikel">
9          <h2>Daftar Artikel</h2>
10         <Post judul="JTI Polinema" isi="Jurusan Teknologi Informasi - Politeknik Negeri Malang"/>
11       </div>
12     )
13   }
14 } export default BlogPost;
```

Setelah itu pada stateless component Post.jsx tangkap parameter yang dilempar oleh statefull component seperti pada Gambar 1.16 dan lihat pada browser apa yang terjadi!.

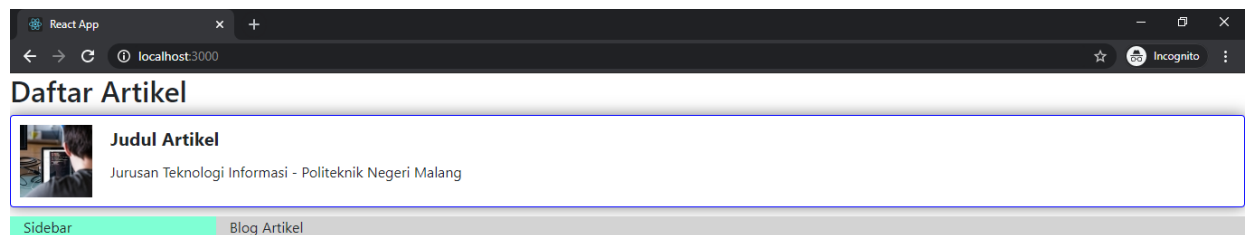


```

framework_prog > 4thWeek > react-web-hunayn > src > component > BlogPost > Post.jsx > Post
1  import React from "react";
2
3  const Post = (props) => {
4    return (
5      <div className="artikel">
6        <div className="gambar-artikel">
7          
8        </div>
9        <div className="konten-artikel">
10         <div className="judul-artikel">Judul Artikel</div>
11         <p className="isi-artikel">{props.isi}</p>
12       </div>
13     </div>
14   )
15 }
16 export default Post;
17
18

```

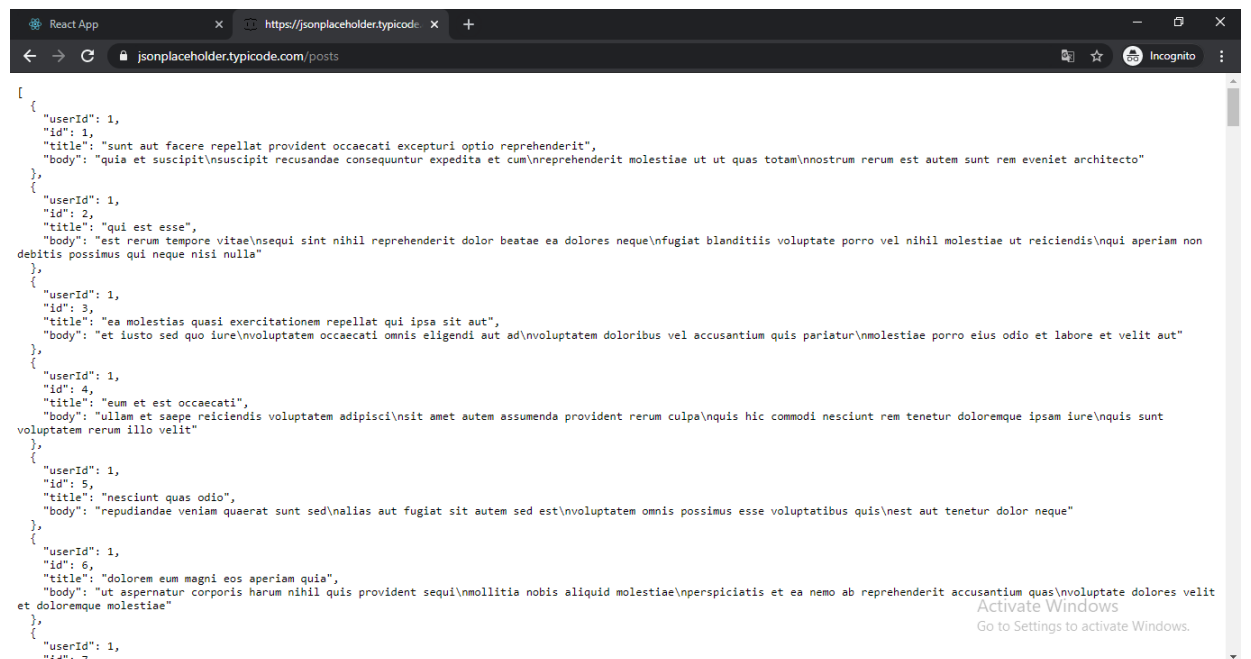
Simpan, dan amati apa yang terjadi pada browser kalian!.



## Mengambil data Post/Artikel dari API.

Gunakan state untuk menyimpan data hasil request dari API

data API yang akan kita gunakan adalah data dummy dari <https://jsonplaceholder.typicode.com/posts>, dimana memiliki 4 element data yaitu userid, id, title, body (seperti pada Gambar 1.17)



Edit pada statefull component BlogPost.jsx seperti pada Gambar 1.18 dan perhatikan dengan seksama akan penjelasan di beberapa baris kode program tersebut.

framework\_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > BlogPost > componentDidMount

```
1 import { Component } from "react";
2 import './BlogPost.css';
3 import Post from "../../component/BlogPost/Post"
4
5 class BlogPost extends Component {
6   state = {                // komponen state dari React untuk stateful component
7     listArtikel: []        // variable array yang digunakan untuk menyimpan data API
8   }
9 }
```

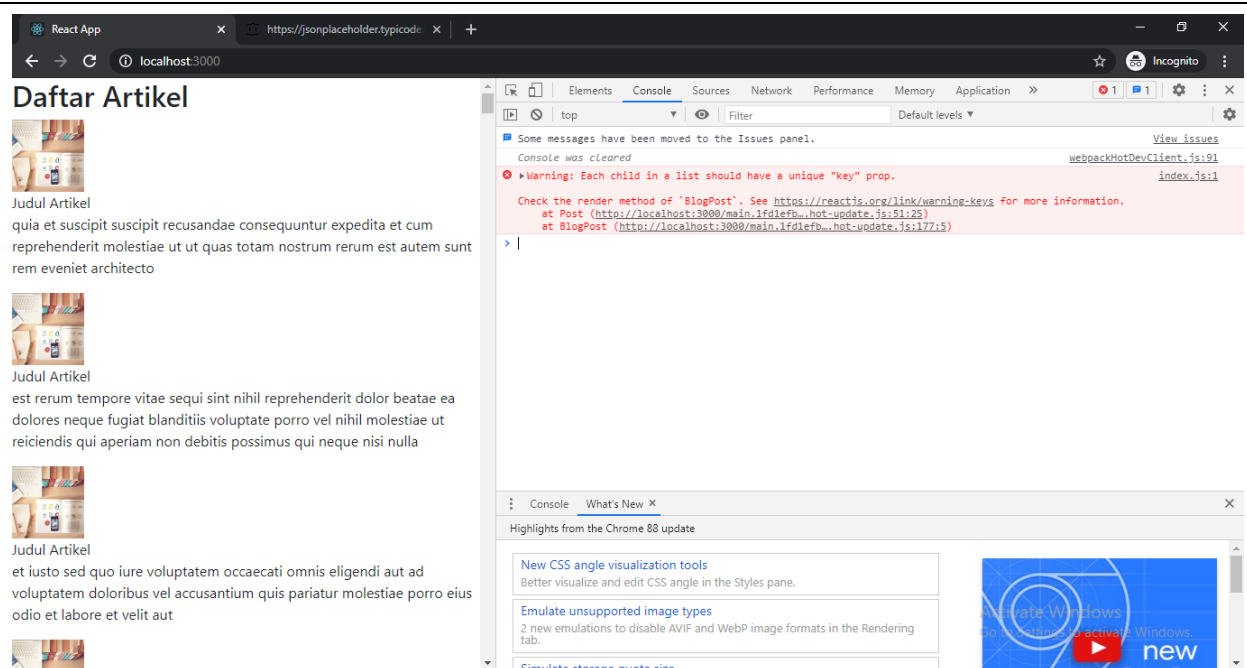
framework\_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > BlogPost > componentDidMount

```
1 import { Component } from "react";
2 import Post from "../../component/BlogPost/Post"
3
4 class BlogPost extends Component {
5   state = {                // komponen state dari React untuk stateful component
6     listArtikel: []        // variable array yang digunakan untuk menyimpan data API
7   }
8
9   componentDidMount() {    // komponen untuk mengecek ketika component telah di-mount-ing, maka panggil API
10     fetch('https://jsonplaceholder.typicode.com/posts') // alamat URL API yang ingin kita ambil datanya
11     .then(response => response.json()) // ubah response data dari URL API menjadi sebuah data JSON
12     .then(jsonHasilAmbilDariAPI => { // data JSON hasil ambil dari API kita masukkan ke dalam listArt
13       this.setState({
14         listArtikel: jsonHasilAmbilDariAPI
15       })
16     })
17   }
18
19   render() {
20     return (
21       <div class="post-artikel">
22         <h2>Daftar Artikel</h2>
23         {
24           this.state.listArtikel.map(artikel => { // looping dan masukkan untuk setiap data yang ada
25             return <Post judul={artikel.title} isi={artikel.body}/> // mappingkan data JSON dari AF
26           })
27         }
28       </div>
29     )
30   }
31 }
```

```
18
19   render() {
20     return (
21       <div class="post-artikel">
22         <h2>Daftar Artikel</h2>
23         {
24           this.state.listArtikel.map(artikel => { // looping dan masukkan untuk setiap data yang ada
25             return <Post judul={artikel.title} isi={artikel.body}/> // mappingkan data JSON dari AF
26           })
27         }
28       </div>
29     )
30   }
31 }
32 export default BlogPost;
```

Lihat hasilnya pada browser. Kemudian klik kanan pada browser pilih "inspect element" kemudian pilih tab "console". Refresh browser dan amati apa yang terjadi

Jika terlihat seperti pada Gambar 1.19, maka terjadi kesalahan pada program yang kita buat



Jika terjadi hal demikian, hal ini terjadi karena dalam react "class" dalam tag html harus ditulis menjadi "className". selain itu, pada statefull component yang dinamis, harus ada "UNIQUE KEY" pada tiap komponen yang diproses sehingga komponen perlu diberi UNIQUE KEY.

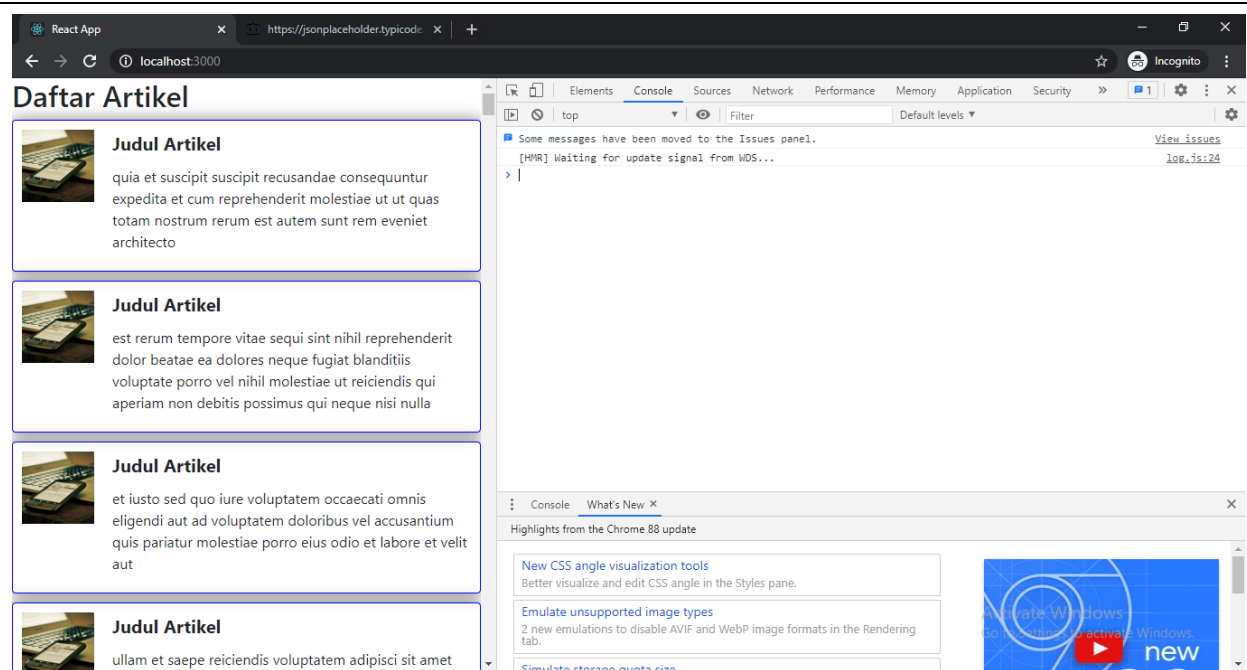
UNIQUE KEY dapat diambil dari element yang ada pada data API yang sudah kita ambil (contoh saat ini adalah element id pada data API (userid, id, title, body) yang akan kita gunakan untuk UNIQUE KEY. Lihat Gambar 1.20.

```

20   render() {
21     return (
22       <div className="post-artikel">
23         <h2>Daftar Artikel</h2>
24         {
25           this.state.listArtikel.map(artikel => { // looping dan masukkan untuk setiap data yang ada
26             return <Post key={artikel.id} judul={artikel.title} isi={artikel.body}/> // mappingkan
27           })
28         }
29       </div>
30     )
31   }
32 }
33 export default BlogPost;

```

Simpan dan lihat apa yang terjadi pada console browser (Gambar 1.21).



### Pertanyaan Praktikum 1

a. Pada langkah 8, sekarang coba kalian ganti class container dengan container-fluid atau sebaliknya pada file "public/index.html" dan lihat apaperbedaannya.

1. Tampilan seperti apa yang kalian temukan setelah mencoba mengganti nama class tersebut?

Apabila class container maka content akan berada di tengah tidak memenuhi ruangan, sedangkan jika class container-fluid content akan memenuhi ruangan hingga ujung kanan kiri full

2. Apa perbedaan dari container dan container-fluid ?

Container untuk mendefinisikan kotak terluar yang menyelimuti row tetapi tidak memenuhi ruangan full kanan kiri sedangkan container-fluid memenuhi ruangan

b. Jika kita ingin meng-import suatu component contoh component bootstrap, akan tetapi component dalam tersebut belum terdapat pada module ReactJS. Apa yang akan dilakukan untuk dapat menggunakan component tersebut? Bagaimana caranya?

Pada root app kita buka melalui terminal kemudian ketikkan "npm install bootstrap" kemudian import dengan syntax "import 'bootstrap/dist/css/bootstrap.min.css';"

### Praktikum 2

#### Interaksi dengan API menggunakan Fake API

Install Fake API (JSON Server)

```

PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn> npm install -gjson-server
npm WARN bootstrap@4.6.0 requires a peer of jquery@1.9.1 - 3 but none is installed. You must install peer dependencies yourself.
npm WARN bootstrap@4.6.0 requires a peer of popper.js@^1.16.1 but none is installed. You must install peer dependencies yourself.
npm WARN tsutils@3.20.0 requires a peer of typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta but none is installed. You must install peer dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.3.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\watchpack-chokidar2\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\webpack-dev-server\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

audited 1942 packages in 18.235s

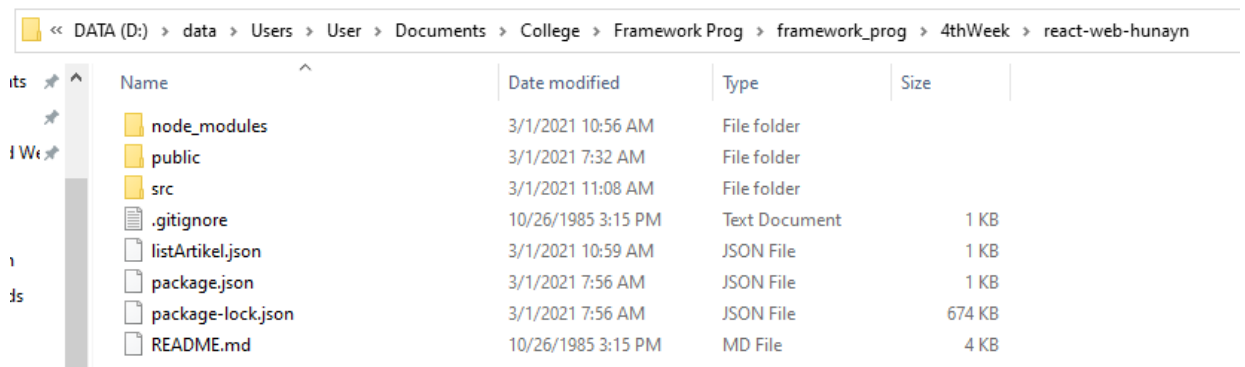
128 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn>

```

Copy-kan file json listArtikel.json yang sudah ada pada direktori project reactjs kita.



Buka cmd baru pada direktori project, lalu ketik perintah json-server --watch listArtikel.json --port 3001

```

PS D:\data\Users\User\Documents\College\Framework Prog\framework_prog\4thWeek\react-web-hunayn> npx json-server --watch -p 3001 listArtikel.json
npx: installed 182 in 10.514s

\(^_^)/ hi!

Loading listArtikel.json
Done

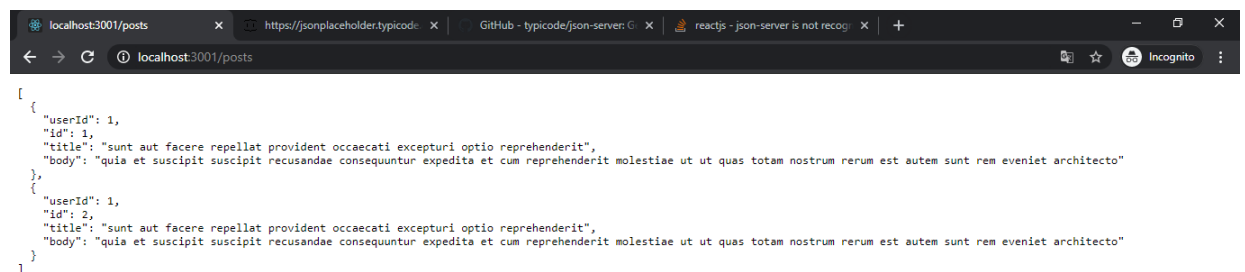
Resources
http://localhost:3001/posts

Home
http://localhost:3001

Type s + enter at any time to create a snapshot of the database
Watching...

```

Kita cek url resource yang ada pada Fake API server ke browser apakah bisa diakses. Ketik url <http://localhost:3001/posts> pada browser



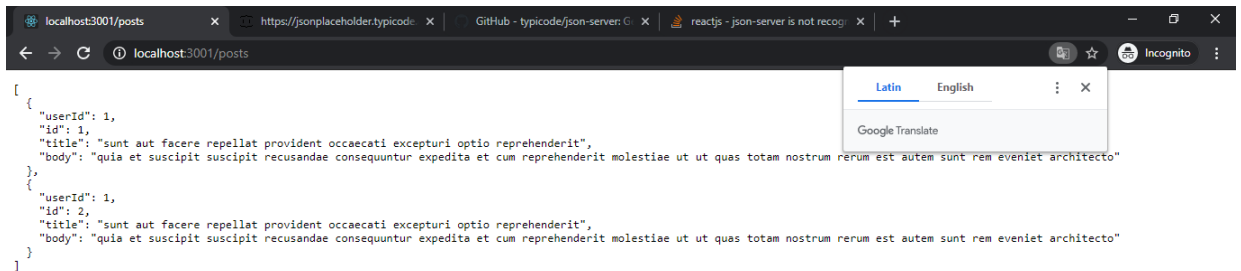
Untuk memastikan lagi, kita edit statefull component BlogPost (Gambar 1.18) pada baris 11. Kita ganti url API dari <https://jsonplaceholder.typicode.com/posts> menjadi <http://localhost:3001/posts>

```

9
10 componentDidMount() { // komponen untuk mengecek ketika component telah di-mount-ing, maka panggil API
11   fetch('http://localhost:3001/posts') // alamat URL API yang ingin kita ambil datanya
12   .then(response => response.json()) // ubah response data dari URL API menjadi sebuah data JSON
13   .then(jsonHasilAmbilDariAPI => { // data JSON hasil ambil dari API kita masukkan ke dalam listArt
14     this.setState({
15       listArtikel: jsonHasilAmbilDariAPI
16     })
17   })
18 }
19

```

Simpan perubahan dan amati apa yang terjadi.



## Pertanyaan Praktikum 2

a. Kenapa json-server dijalankan pada port 3001? Kenapa tidak sama-sama dijalankan pada port 3000 seperti project react yang sudah kita buat?

Karena port 3000 sudah digunakan untuk operasi GET pada <https://jsonplaceholder.typicode.com/posts> sedangkan untuk mengambil data di local project react-web-hunayn harus menggunakan port yang berbeda

b. Bagaimana jadinya kalau kita ganti port json-server menjadi 3000?

Bisa diganti ke port 3000 untuk mengambil data local yang menjadi server tetapi job yang sebelumnya harus dilakukan terminate kemudian run pada port 3000

## Praktikum 3

### Interaksi dengan API menggunakan method DELETE

Buka stateless component Post. Tambahkan 1 baris kode program pada baris 10 seperti pada Gambar 3.1

```

framework_prog > 4thWeek > react-web-hunayn > src > component > BlogPost > Post.jsx > Post
1  import React from "react";
2
3  const Post = (props) => {
4    return (
5      <div className="artikel">
6        <div className="gambar-artikel">
7          
8        </div>
9        <div className="konten-artikel">
10         <div className="judul-artikel">Judul Artikel</div>
11         <p className="isi-artikel">{props.isi}</p>
12         <button className="btn btn-sm btn-warning" onClick={() => props.hapusArtikel(props.idArtikel)}>Hapus</button>
13       </div>
14     </div>
15   )
16 }
17 export default Post;
18
19

```

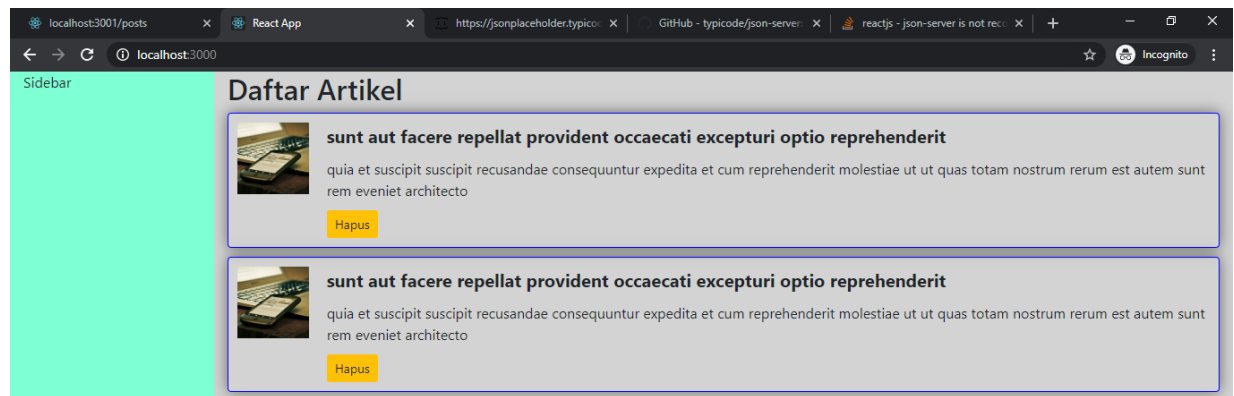
Kemudian pada statefull component BlogPost, modifikasi kode program sebelumnya sesuai dengan Gambar 3.2

```

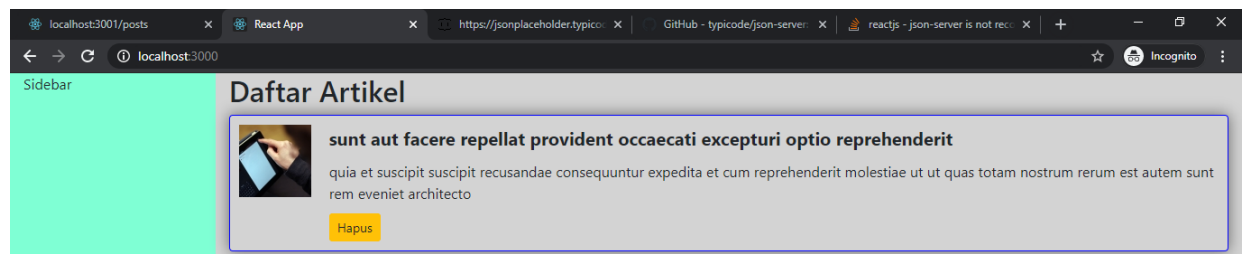
19
20   handleHapusArtikel = (data) => {
21     fetch('http://localhost:3001/posts/${data}', {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS
22     .then(res => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
23       this.ambilDataDariServerAPI()
24     })
25   }
26
27   render() {
28     return (

```

Klik tombol hapus pada list artikel di browser. Amati apa yang terjadi



Ketika tombol dihapus maka index tersebut akan terhapus sehingga hanya menampilkan index yang tersisa



### Pertanyaan Praktikum 3

- Apa yang terjadi setelah kalian klik tombol hapus?  
Maka data yang diklik hapus akan hilang, sehingga hanya menampilkan data yang tersisa
- Perhatikan file listArtikel.json, apa yang terjadi pada file tersebut? Kenapa demikian?  
Data pada id = 1 terhapus karena telah di klik button hapus pada id tersebut.
- Fungsi handleHapusArtikel itu untuk apa?  
Function yang digunakan untuk menghandle dalam penghapusan data, dimulai dengan fetch data pada localhost:3001/posts kemudian menjalankan method 'DELETE'
- Jelaskan perbedaan fungsi componentDidMount() pada Gambar 1.18 dengan fungsi componentWillUnmount() pada Gambar 3.2 ?  
componentDidMount() pada gambar 1.18 digunakan untuk mengambil data pada <https://jsonplaceholder.typicode.com/posts> sedangkan pada gambar 3.2 digunakan untuk mengambil data pada server local

### Praktikum 4

#### Interaksi dengan API menggunakan method POST

Buka statefull component BlogPost, dan modifikasi pada fungsi render() untuk menampilkan form input artikel yang berisi judul dan isi berita. seperti pada Gambar 4.1

```

60
61 render() {
62   return (
63     <div className="post-artikel">
64       <div className="form pb-2 border-bottom">
65         <div className="form-group row">
66           <label htmlFor="title" className="col-sm-2 col-form-label">Judul</label>
67           <div className="col-sm-10">
68             <input type="text" className="form-control" id="title" name="title" onChange={this.handleTambahArtikel}/>
69           </div>
70         </div>
71         <div className="form-group row">
72           <label htmlFor="body" className="col-sm-2 col-form-label">Isi</label>
73           <div className="col-sm-10">
74             <textarea className="form-control" id="body" name="body" rows="3" onChange={this.handleTambahArtikel}></textarea>
75           </div>
76         </div>
77         <button type="submit" className="btn btn-primary" onClick={this.handleTombolSimpan}>Simpan</button>
78       </div>
79       <h2>Daftar Artikel</h2>
80       {
81         this.state.listArtikel.map(artikel => { // looping dan masukkan untuk setiap data yang ada di listArtikel ke variable artikel
82           return <Post key={artikel.id} judul={artikel.title} isi={artikel.body} idArtikel={artikel.id} hapusArtikel={this.handleHapusArtikel}/> // mappingkan data JSON
83         })
84       }
85     </div>
86   )
87 }
88
89 export default BlogPost;

```

Kemudian modifikasi BlogPost untuk bagian state dan request API dari server, seperti Gambar 4.2

```

framework_prog > 4thWeek > react-web-hunayn > src > container > BlogPost > BlogPost.jsx > BlogPost > render
1  import { Component } from "react";
2  import './BlogPost.css';
3  import Post from "../../component/BlogPost/Post"
4
5  class BlogPost extends Component {
6    state = {
7      listArtikel: [], // komponen state dari React untuk stateful component
8      insertArtikel: { // variable array yang digunakan untuk menyimpan data API
9        userId: 1, // variable yang digunakan untuk menampung sementara data yang akan di insert
10        id: 1, // kolom userId, id, title, dan body sama, mengikuti kolom yang ada pada listArtikel.json
11        title: "",
12        body: ""
13      }
14    }
15
16    ambilDataDariServerAPI() { // fungsi untuk mengambil data dari API dengan penambahan sort dan order
17      fetch('http://localhost:3001/posts?_sort=id&_order=desc') // penambahan sort dan order berdasarkan parameter
18      .then(response => response.json()) // ubah response data dari URL API menjadi sebuah data JSON
19      .then(jsonHasilAmbilDariAPI => { // data JSON hasil ambil dari API kita masukkan ke dalam listArtikel pada state
20        this.setState({
21          listArtikel: jsonHasilAmbilDariAPI
22        })
23      })
24    }
25
26    componentDidMount() { // komponen untuk mengecek ketika component telah di-mount-ing, maka panggil API
27      this.ambilDataDariServerAPI() // ambil data dari server API lokal
28    }
29

```

Tambahkan untuk handle form tambah data artikel seperti Gambar 4.3

```

29
30 handleHapusArtikel = (data) => { // function yang meng-handle button action hapus data
31   fetch('http://localhost:3001/posts/${data}', {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS datanya
32   .then(res => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
33     this.ambilDataDariServerAPI()
34   })
35 }
36
37 handleTambahArtikel = (event) => { // fungsi untuk meng-handle form tambah data artikel
38   let formInsertArtikel = {...this.state.insertArtikel}; // cloning data state insertArtikel ke dalam variable formInsertArtikel
39   let timestamp = new Date().getTime(); // digunakan untuk menyimpan waktu (sebagai ID artikel)
40   formInsertArtikel['id'] = timestamp;
41   formInsertArtikel[event.target.name] = event.target.value; // menyimpan data onChange ke formInsertArtikel sesuai dengan target yang di
42   this.setState({
43     insertArtikel: formInsertArtikel
44   });
45 }
46
47 handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
48   fetch('http://localhost:3001/posts', {
49     method: 'post', // method POST untuk input/insert data
50     headers: {
51       'Accept': 'application/json',
52       'Content-Type': 'application/json'
53     },

```

Langkah terakhir tambahkan fungsi untuk handle tombol simpan artikel, seperti pada Gambar 4.4

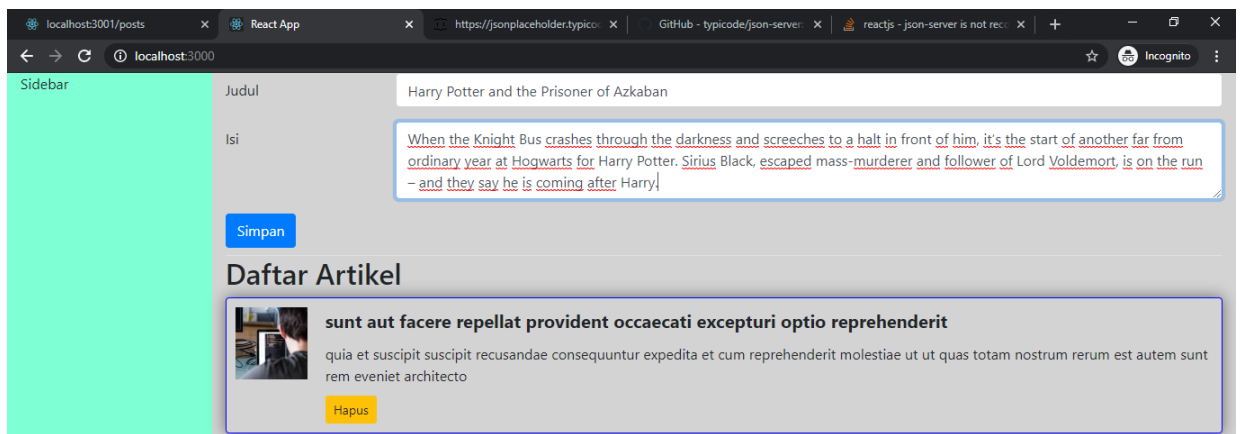
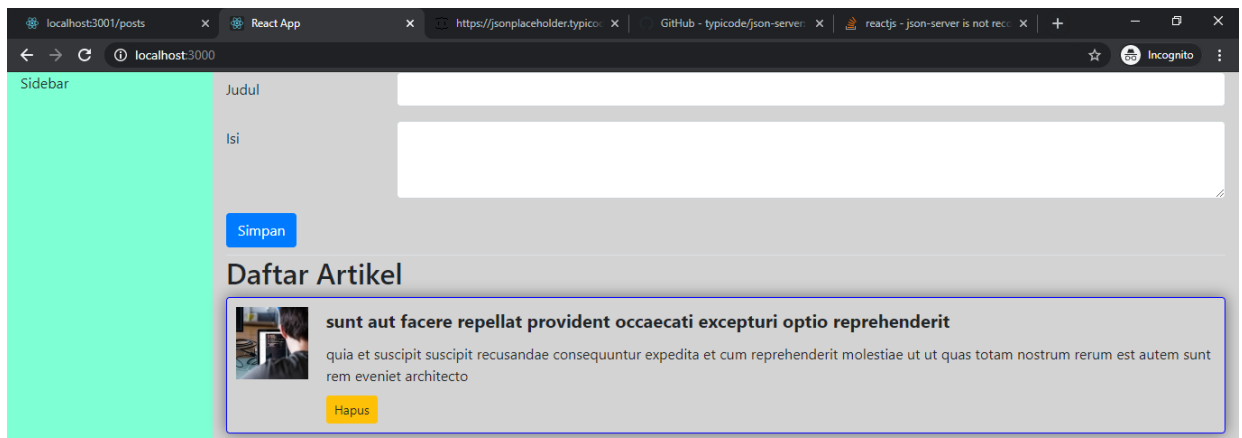


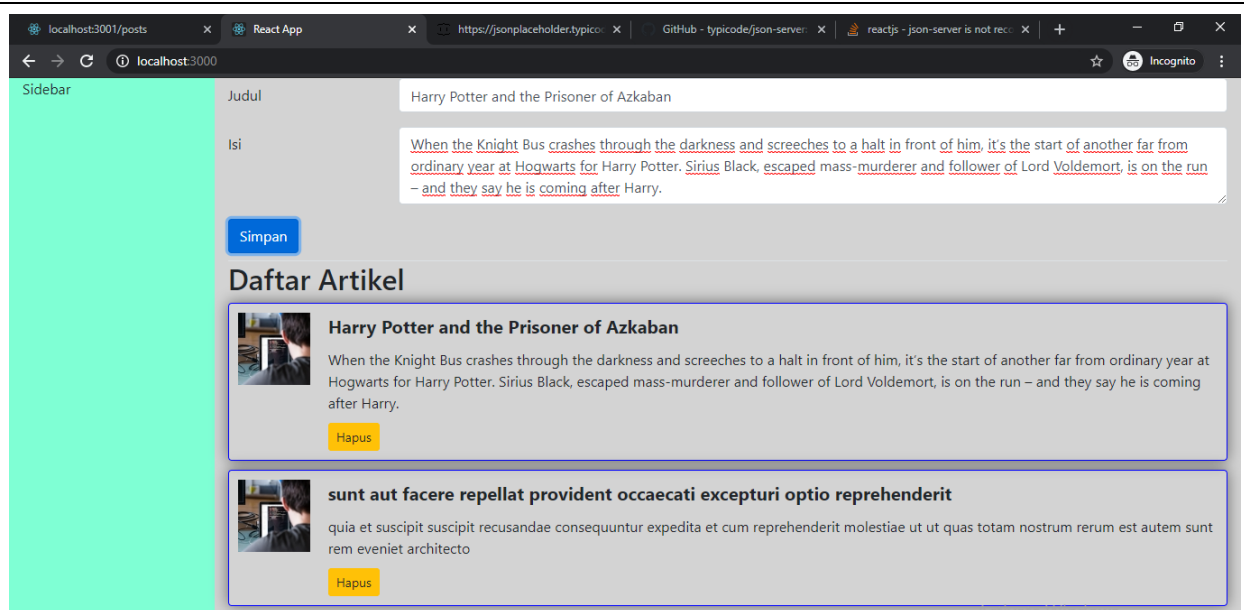
```

44     });
45   }
46
47   handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
48     fetch('http://localhost:3001/posts', {
49       method: 'post', // method POST untuk input/insert data
50       headers: {
51         'Accept': 'application/json',
52         'Content-Type': 'application/json'
53       },
54       body: JSON.stringify(this.state.insertArtikel) // kirimkan ke body request untuk data yang akan ditambahkan (insert)
55     })
56     .then((Response) => {
57       this.ambilDataDariServerAPI(); // reload/refresh data
58     });
59   }
60
61   render() {
62     return (
63       <div className="post-artikel">
64         <div className="form pb-2 border-bottom">

```

Simpan, lakukan percobaan penambahan data, dan amati perubahannya



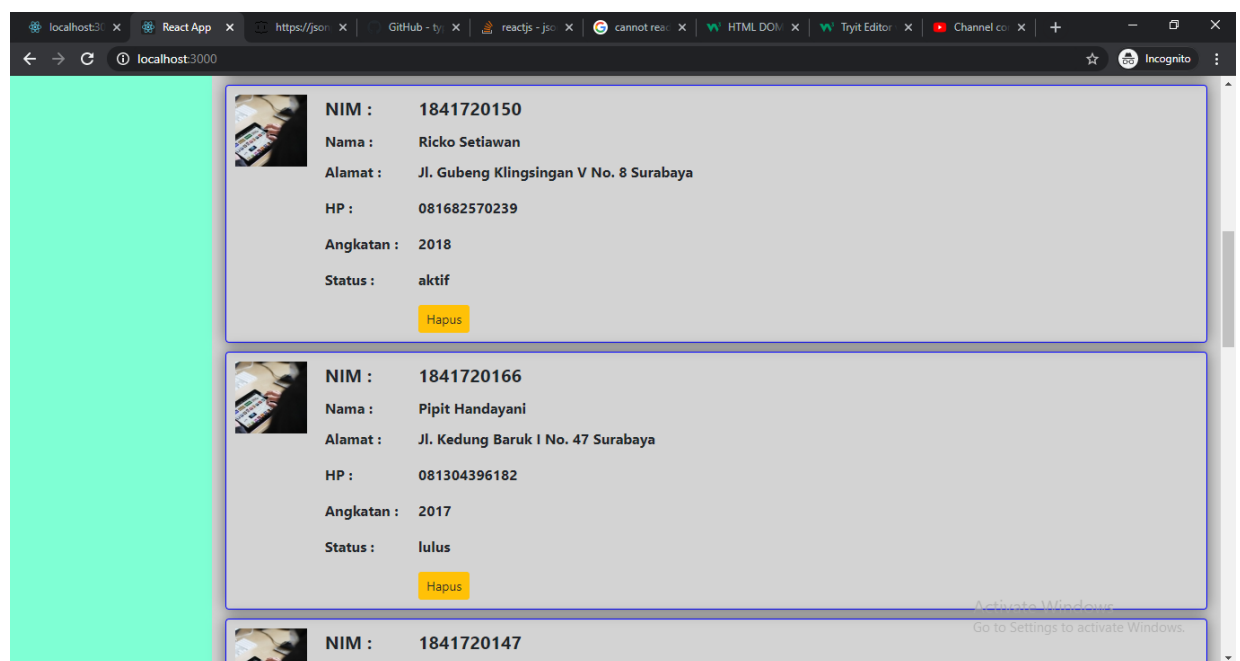
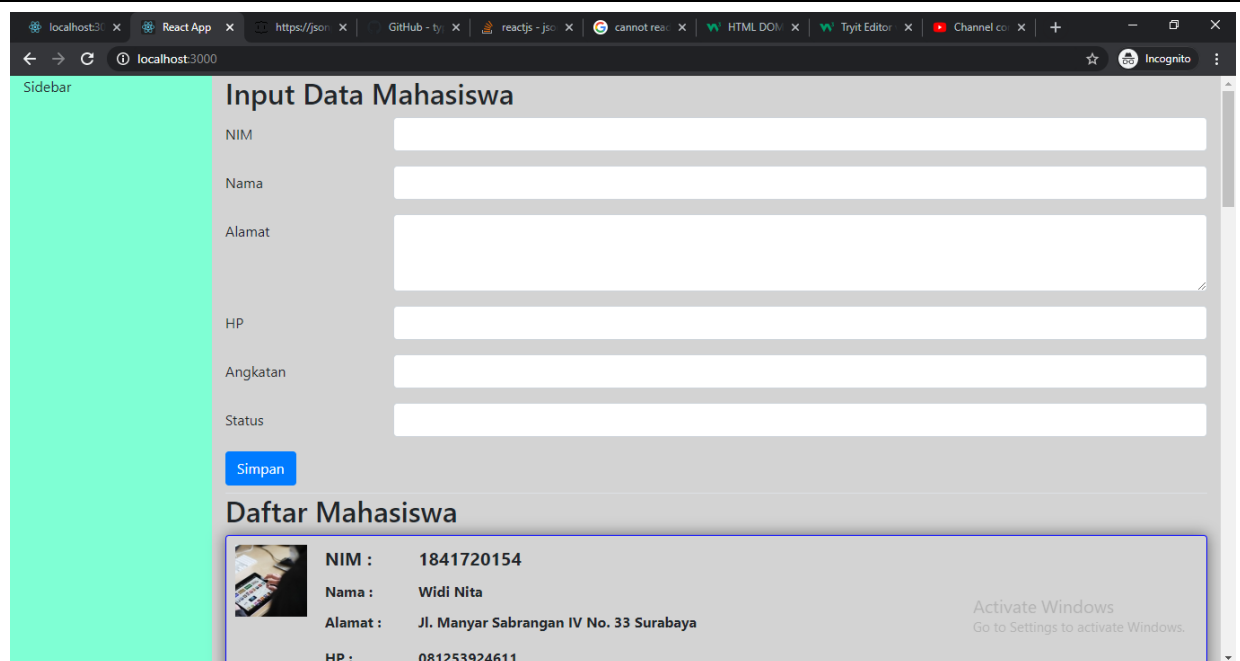


#### Pertanyaan Praktikum 4

- Jelaskan apa yang terjadi pada file listArtikel.json sebelum dan setelah melakukan penambahan data?  
Sebelum dilakukan penambahan data, file listArtikel.json dilakukan pengambilan data terlebih dahulu kemudian dilakukan sorting dan order secara desc lalu data diubah menjadi JSON kemudian data JSON tersebut dilakukan insert ke listArtikel pada state. Setelah penambahan data, data akan ditampilkan sementara dalam insertArtikel kemudian dilakukan eksekusi function handleTambahArtikel, dan terakhir eksekusi function handleTombolSimpan
- Data yang ditampilkan di browser adalah data terbaru berada di posisi atas dan data lama berada di bawah, sedangkan pada file listArtikel.json data terbaru malah berada di bawah. Jelaskan mengapa demikian?  
Karena pada saat function ambilDataDariServerAPI() dieksekusi dilakukan order berdasarkan id secara desc sehingga data dengan id yang lebih besar (terakhir dilakukan input) akan berada pada posisi paling atas diikuti dengan id yang lebih kecil berada di bawahnya

#### TUGAS PRAKTIKUM

Buatlah program menggunakan Fake API (JSON Server) tentang pendataan Mahasiswa aktif/cuti/lulus di Jurusan Teknologi Informasi. Atribut-atribut yang ada dari mahasiswa adalah NIM, nama, alamat, no hp, tahun Angkatan, dan status. Buatlah aplikasi yang menggunakan API dengan method GET, DELETE, dan POST.



Link GitHub : [https://github.com/hunaynr/framework\\_prog/tree/main/4thWeek](https://github.com/hunaynr/framework_prog/tree/main/4thWeek)

Link Youtube : <https://youtu.be/Dq8-0gz1W1o>



